GCCS/DII COE System Integration Support

DII COE Segment System Requirement Specifications (for the SeComp Tool Version 1.0.0.3)

May 21, 1997

Prepared for:

DISA/JEJA
ATTN: Claire Burchell
45335 Vintage Park Plaza
Sterling, VA 20166-6701

Contract Number: DCA100-94-D-0014
Delivery Order Number: 330, Task 6
CDRL Number: A005

Prepared by:

Computer Sciences Corporation
Defense Enterprise Integration Services
Four Skyline Place
5113 Leesburg Pike, Suite 700
Falls Church, VA 22041

THIS DOCUMENT IS UNCLASSIFIED

Defense Information Infrastructure (DII)

Common Operating Environment (COE)

Segment System Requirement Specifications for the Security Compliance (SeComp) Tool Version 1.0.0.3

May 21, 1997

Prepared for:

DISA/JEJA
ATTN: Claire Burchell
45335 Vintage Park Plaza
Sterling, VA 20166-6701

Prepared by:

Computer Sciences Corporation
Defense Enterprise Integration Services
Four Skyline Place
5113 Leesburg Pike, Suite 700
Falls Church, VA 22041

Table of Contents

LIST OF FIGURES

LIST OF TABLES

Preface

The following conventions are used in this document:

Bold	Used for information that is typed, pressed, or selected in executables and instructions. For example, select **connect to host**.
Italics	Used for file names, directories, scripts, commands, user IDs, document names, and Bibliography references; and any unusual computerese the first time it is used in text.
Underline	Used for emphasis.
Arrows <>	Used to identify keys on the keyboard. For example <Return>.
"Quotation Marks"	Used to identify informal, computer-generated queries and reports, or coined names; and to clarify a term when it appears for the first time. For example "Data-Generation Report."
`Courier Font`	Used to denote anything as it appears on the screen or command lines. For example `tar xvf dev/rmt/3mm`.
Capitalization	Used to identify keys, screen icons, screen buttons, field, and menu names.

1. System Requirement Specifications (SRS) for the SeComp Tool

The Defense Information Infrastructure (DII) Common Operating Environment (COE) Security Compliance (SeComp) tool provides functions and procedures to determine if, where, when, and how COE kernel function and application segments might be disruptive to a system security configuration. The SeComp tool examines and reports on the configuration items identified in the DII COE Security Checklist.

2. Software Design

The SeComp tool is designed to report the security-relevant configuration details of a UNIX-based operating system (OS) and associated file system content. The tool interrogates and reports both file system statistical and content information.

The SeComp tool is a command line, command driven tool. This means:

- That all SeComp operations are entered at the command line, and
- All runtime configuration details are capable of being defined in the command line, including task definition, phase selection, report directory location, and master directory location.

The SeComp tool is configuration driven. This means:

- That the SeComp tool will execute from anywhere on the host file system,
- The SeComp reports and master files may be directed to locations outside of the runtime software location, and
- Task and some implementation values may be tailored by modifying configuration variables located in the SeComp configuration files.

All configuration variables that are recommended for modification are located in the first level */etc/secomp* hierarchy files. Modifying variables in the programs below the first level hierarchy files puts the SeComp operational design at risk and should be avoided.

The SeComp tool has a non-verbose and verbose option. This means that the report output may be made verbose or non-verbose. Verbose operations will supply a great deal of security knowledge information within the context of the report output. Non-verbose mode sticks to just reporting the information and is the default mode of operation.

The SeComp tool is designed to operate in a completely passive manner. The only invasive results to the host file system are in the reports that are generated as SeComp executes.

The SeComp tool is written completely in scripting language to allow non-compiled portability from UNIX platform to UNIX platform. The tool is written primarily in System V Release 4 UNIX system variants for the Bourne shell, however, there are modifications required when moving SeComp software across vendor platforms due to proprietary command differences.

The software structure is described in the following paragraphs.

2.1 Client/Server Architecture

The SeComp tool may be executed on multiple clients simultaneously with each instantiation in isolation of all others. Only one copy of the SeComp software needs to exist in the network domain, and for the purpose of this document, will be referred to as the SeComp server. The SeComp server will contain the SeComp programs, and by default, will house the SeComp master and report file.

The SeComp clients connect to the SeComp server using the NFS *automount* utility. The *automount* utility maintains an NFS connection for only as long as the connection is required and therefore contributes to a reduction in network burden. The details of the SeComp automount configuration are referenced in the *DII COE SeComp System Administrators Manual* (SAM).

All SeComp generated *reports* and *master* files bear a date stamp and the name identifying the time the reports were generated and the host name of the system that the reports represent. All generated *report* and *master* files for a host name are stored in a hierarchy specifically created by the SeComp programs for that host, by the host name. Additionally, each set of reports are held in a directory that is identified by the time stamp that represents the SeComp program run time.

2.2 File System Hierarchy

The SeComp file system hierarchy is described in Figure 1.

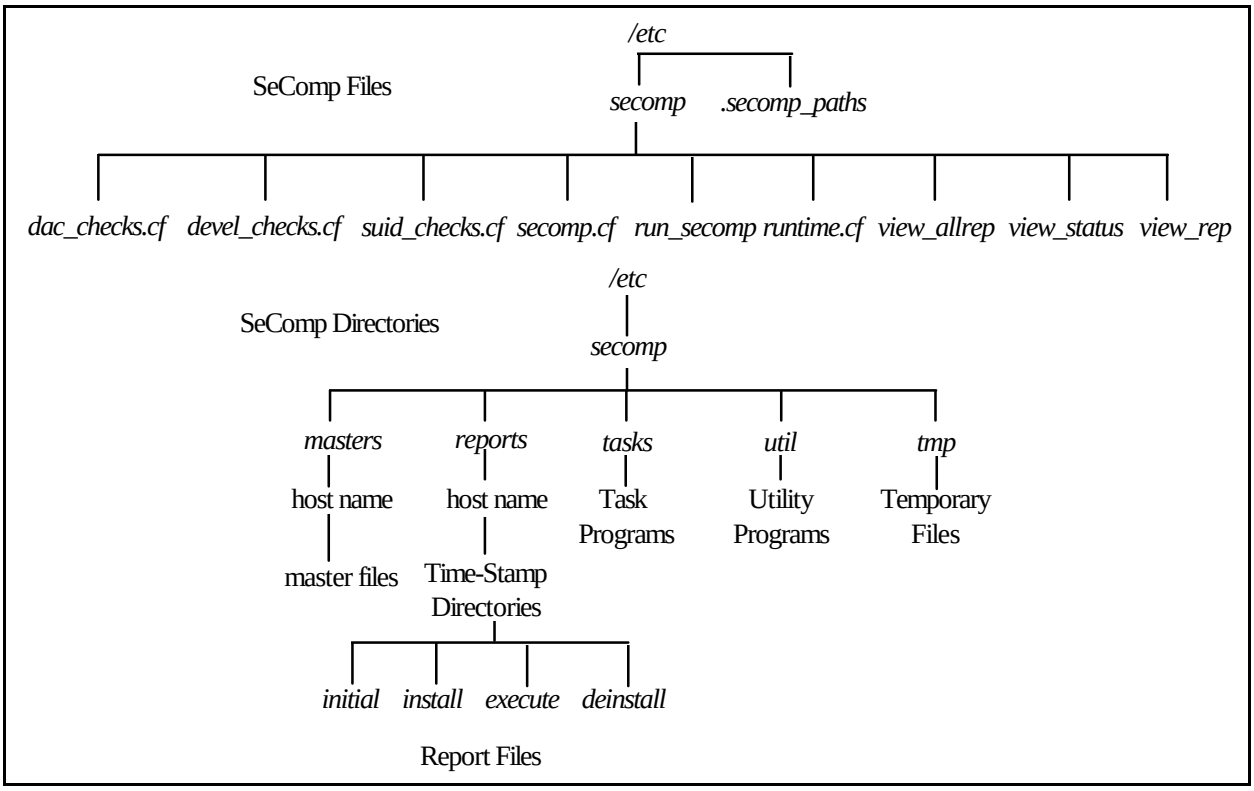

Figure 1. SeComp File System and Directory Structure

The SeComp main functions will cause the execution of the SeComp primary tasks that check the systems security configuration in the primary task areas of:

- Identification and Authentication (I&A)
- Discretionary Access Control (DAC)
- Audit
- System Configuration.

The primary tasks will cause the execution of the SeComp subordinate tasks that execute the various checking programs on behalf of the primary task areas. The primary task files are located in the */etc/secomp/tasks* directory and are suffixed with *.task*. Each task file contains calls to the subordinate task programs that are also located in the */etc/secomp/tasks* directory.

The subordinate task programs may execute UNIX commands, other subordinate programs, or SeComp program utilities. The SeComp utility programs are located in the */etc/secomp/util* directory.

2.3 SeComp Configuration Files

All applicable SeComp configuration files are located in the */etc/secomp* hierarchy. The primary SeComp configuration files are:

- *runtime.cf*
- *secomp.cf*.

In addition to these configuration files, there are other configuration files that will be suffixed with *.cf* that are used to assign implementation-specific values that may used to tailor the SeComp program to the system requirements.

2.3.1 The *runtime.cf* File

The main functions understand the secomp environment structure through the consistent referencing of pathing and other configuration variables that are assigned in the *runtime.cf* configuration file. These values are exported and used by all primary and subordinate task programs and configuration files. The primary pathnames assigned in the *runtime.cf* file are:

- *TASKS*: The SeComp task programs to be executed. Defaults to the *PRE*, *audit*, *sys_config*, *INA*, *DAC*, and *sys_share* tasks (all of the possible tasks).

- *SECOMP_REPORTDIR*: The location of the SeComp reports hierarchy -- defaults to *${SECOMPDIR}/reports*.

- *SECOMP_MASTERSDIR*: The location of the SeComp master files hierarchy -- defaults to *${SECOMPDIR}/masters*.

The *SECOMPDIR* variable is always assigned to the *pwd* command output by the *run_secomp* program file and exported there also. This means that *SECOMPDIR* is assigned to the pathname where the *run_secomp* program file is located.

The *runtime.cf* file may be used to assign these locations for convenience. These task and locations may also be assigned through the use of command line options.

The *runtime.cf* file also contains variables that may be assigned that tune the SeComp report out more to the system security policy. These variables are shown with their current default assignments:

```
SYSNAME="GCCS"
UVAL="002"
UMASK_CENTRAL=/h/.umasks
HOMEDIRBITS=0775
MIN_AUDIT_FLAGS="lo pe -fc -fd -pr -fw"
NETIFACE="lan"
MAXIFACEID=10
```

The *NETIFACE* and *MAXIFACEID* are used in the HP-UX SeComp version to identify a range and name for the network interface card. This is used to determine network interface identification used to get the interface status information.

2.3.2 The *secomp.cf* File

The *secomp.cf* file is the primary configuration file for the SeComp programs. The *secomp.cf* file contains configuration details for the SeComp programs in general, and some implementation-specific value assignments.

One very important role of the *secomp.cf* file is to create the directories and hierarchy required for the generation of the SeComp reports.

The *secomp.cf* file identifies the pathname locations of all of the system commands that are used by the SeComp primary and subordinate task programs. Additionally, the *secomp.cf* file validates the existence of the commands on the file system, in the location specified. Finally, the *secomp.cf* file exports the command-sourcing variables.

The *secomp.cf* file provides another important service by establishing which, if any, network information system (NIS or NIS+ - Solaris only) that is being used by the system under test. Variables are assigned and exported to identify this status to the task programs.

2.4 Crack Programs

The Crack password vulnerability program is integrated into the SeComp tool. The Crack program will attempt to break each password it finds on the system. This is not a SeComp program, however, a SeComp front-end program was developed and is in place to automate the Crack program execution. Crack report files are directed to the same reports hierarchy established for the SeComp programs. The front-end will check the system for evidence of NIS, NIS+, or local password file use, or the logical combinations of these files and feed the entire password content to the Crack programs. The Crack programs may be accessed via the world-wide web (WWW) if additional details are required.

3. Main Functions

The SeComp main program and function is contained in the *run_secomp* program.

3.1 The *run_secomp* File

The *run_secomp* program is the main SeComp program. The *run_secomp* program will cause the execution of the tasks and invoke the runtime options that have been selected. The locations of the runtime, reports, and masters directories are defined in the *run_secomp* program and it exports the locations to the primary and subordinate task programs.

The *run_secomp* program will generate an execution report on behalf of the host named execution called *secomp_exec.log*. The execution report will be located in the host named reports hierarchy.

3.2 SeComp Phase Execution

The *run_secomp* program parses the command line executed and determines the SeComp phase to be entered. The SeComp phases and command line options are shown in Table 1.

Table 1. SeComp Phases and Command Line Options

Phase	**Option**	**Generates**
Initial	*-b*	**.rpt, taskstatus, secomp_exec.log* files
Install	*-I*	**.rpt, *.diff, taskstatus, secomp_exec.log* files
Execute	*-e*	**.rpt, *.diff, taskstatus, secomp_exec.log* files
De-install	*-z*	**.rpt, *.diff, taskstatus, secomp_exec.log* files

The *secomp.cf* file creates the necessary phase execution directory and focuses the report generation for that phase to its phase execution directory.

The initial phase generates all of the master files and initial SeComp task reports. The master files are not modified by any of the remaining phases. In fact, the only method of modifying the master files for a given host name execution is to delete that host's master file(s).

The install, execute, and de-install phase executions automatically generate a UNIX *diff* between the phase execution report files and the initial phase execution files upon completion of the install, execute, or de-install cycle executions. The *diff* files generated for a phase are deposited in the phase execution report directory (i.e., execute phase diff files end up in */etc/secomp/reports/hostname/latest/execute/*.diff*).

4. SeComp Runtime Program Flow

The SeComp program flow results begins by invoking the *secomp.cf* and *runtime.cf* configuration programs. The *runtime.cf* program assigns the runtime tasking and location variables while the *secomp.cf* program assigns the program internal variables. Figure 2 illustrates the programmatic flow of the SeComp programs.

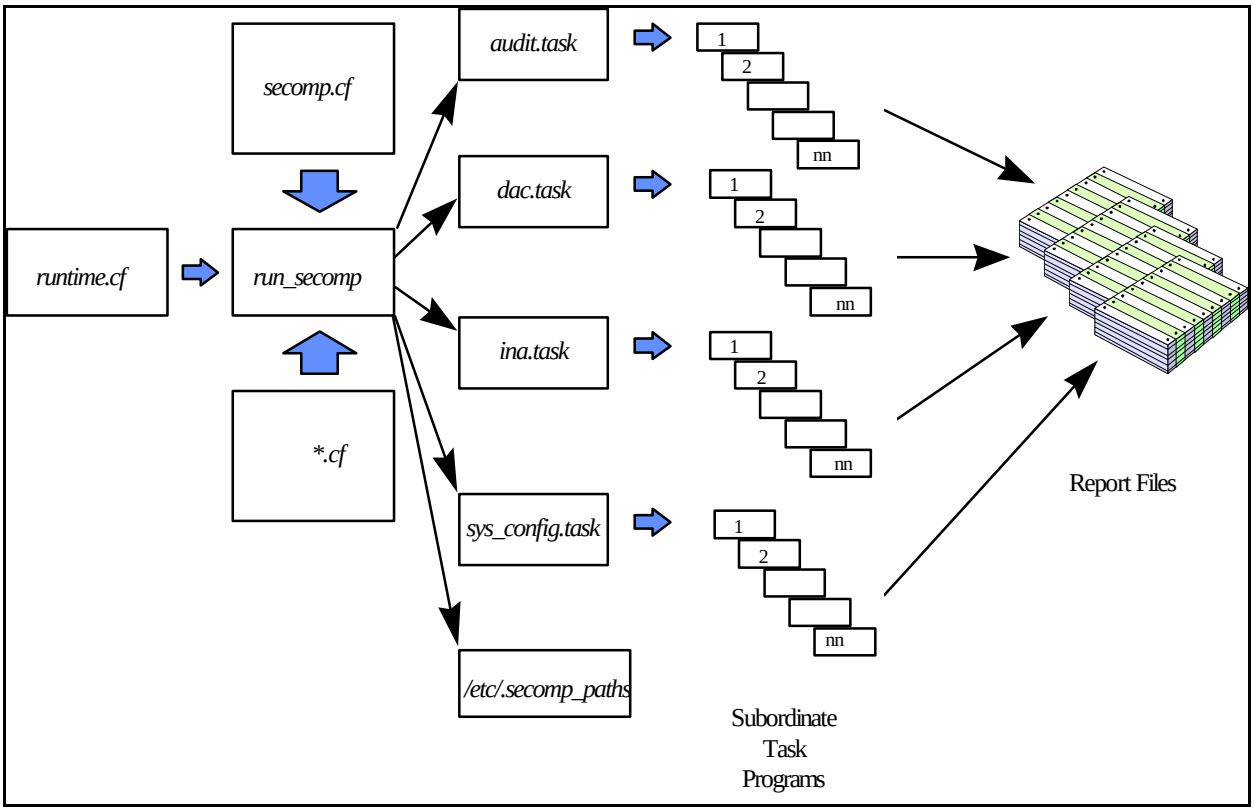

Figure 2. SeComp Programmatic Flows

The *run_secomp* program identifies the tasking and location variable settings and uses those settings to execute the primary tasking (I&A, DAC, System Configuration, and Audit) programs in a background process. The *run_secomp* program will also assign and export the implementation-specific configuration variables.

The primary tasking programs will serially execute each of the subordinate tasking programs identified in the primary tasking program. All task programs are executed as a background process. Each subordinate task program will execute the programs identified within its program. Each subordinate task program directs its output into the report or master file on behalf of the host system on which it is executing.

Appendix A. References

- *Defense Information Systems Agency (DISA), Security Checklists for the Defense Information Infrastructure (DII) Common Operating Environment (COE),* November 1996.

- *Defense Information Infrastructure (DII) Common Operating Environment (COE) Security Compliance (SeComp) Tool System Administrators Manual (SAM), Version 1.0.0.3,* May 21, 1997.